To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

Renesas Electronics website: http://www.renesas.com

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to http://www.renesas.com/inquiry.

RENESAS

H8SX Series

Pulse Output (16-bit Timer)

Introduction

As well as having an architecture that is upward-compatible with each CPU of the H8/300, H8/300H, and H8S series, so as to inherit a full complement of peripheral functions, the H8SX microcomputer series has a maximum operating frequency of 50 MHz and uses a 32-bit H8SX core CPU as well as an on-chip multiplier/divider to improve performance.

This H8SX series Application Note provides information you may be need during software and hardware design. This is a basic edition that provides operation examples that each use a single H8SX series on-chip peripheral function.

Although the operation of each program, circuit, and other aspects covered by this application note has been checked, make sure that you conduct your own operation checks before actually using the H8SX series.

Contents

1. Overview

The 16-bit timer pulse unit of the H8SX series outputs pulses at fixed intervals.

2. Configuration

The example shown below uses channel 0 of the 16-bit timer pulse unit (TPU) to output pulses from the output compare output pin (TIOCA0) at a 50% duty cycle. When the peripheral module clock (Pϕ) is 25 MHz, you can set any output pulse cycle from 83.33 nsec[1)] to 174.76 msec[2)]. Figure 1 is a block diagram of the pulse output.

[1)] When the count clock is Pϕ/1 and the timer general register (TGR) is set to 0 × 0001 (minimum value)

[2)] When the count clock is Pϕ/64 and the timer general register (TGR) is set to 0 × FFFF (maximum value)

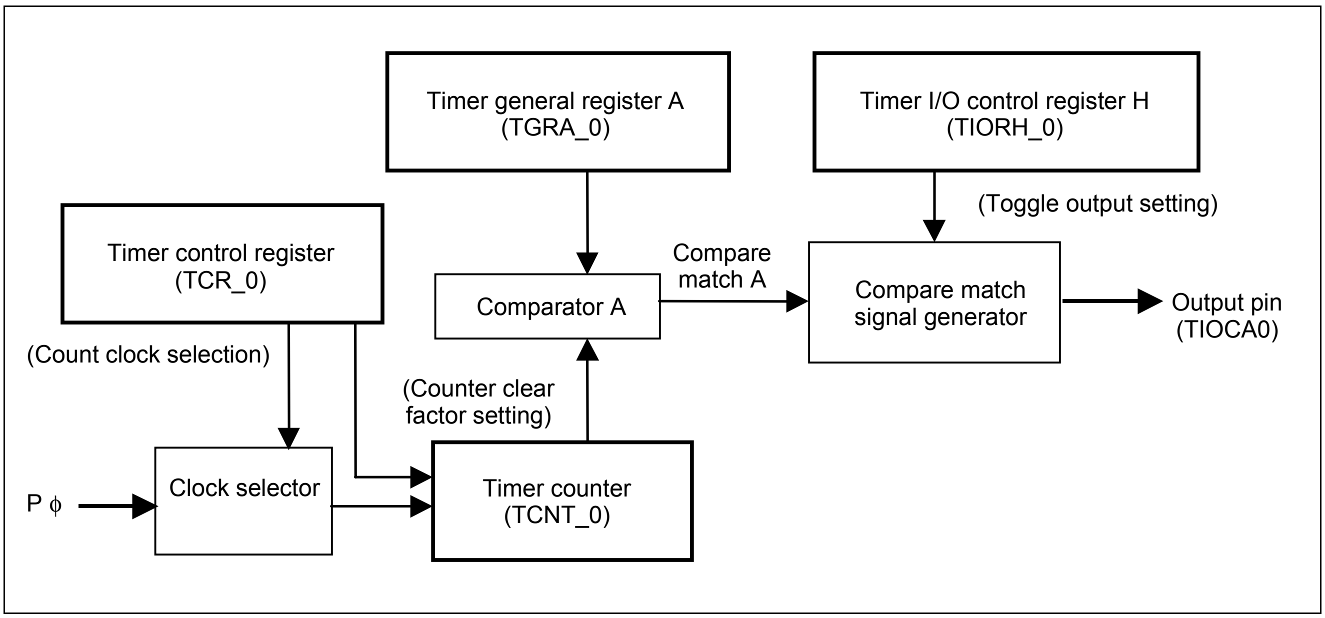


Figure 1 Block Diagram of Pulse Output

3. Sample Program

3.1 Function

This sample program outputs pulses according to the timer value for the pulse cycle. Assume that the duty cycle of the output pulse is 50%. You can calculate the timer value for the pulse cycle by using the following equation:

pulse-cycle = timer-value × count-clock

Also, assume that the count clock is peripheral module (Pϕ)/1. When Pϕ is 25 MHz, the count clock is 40 nsec. Figure 2 shows an operation example.

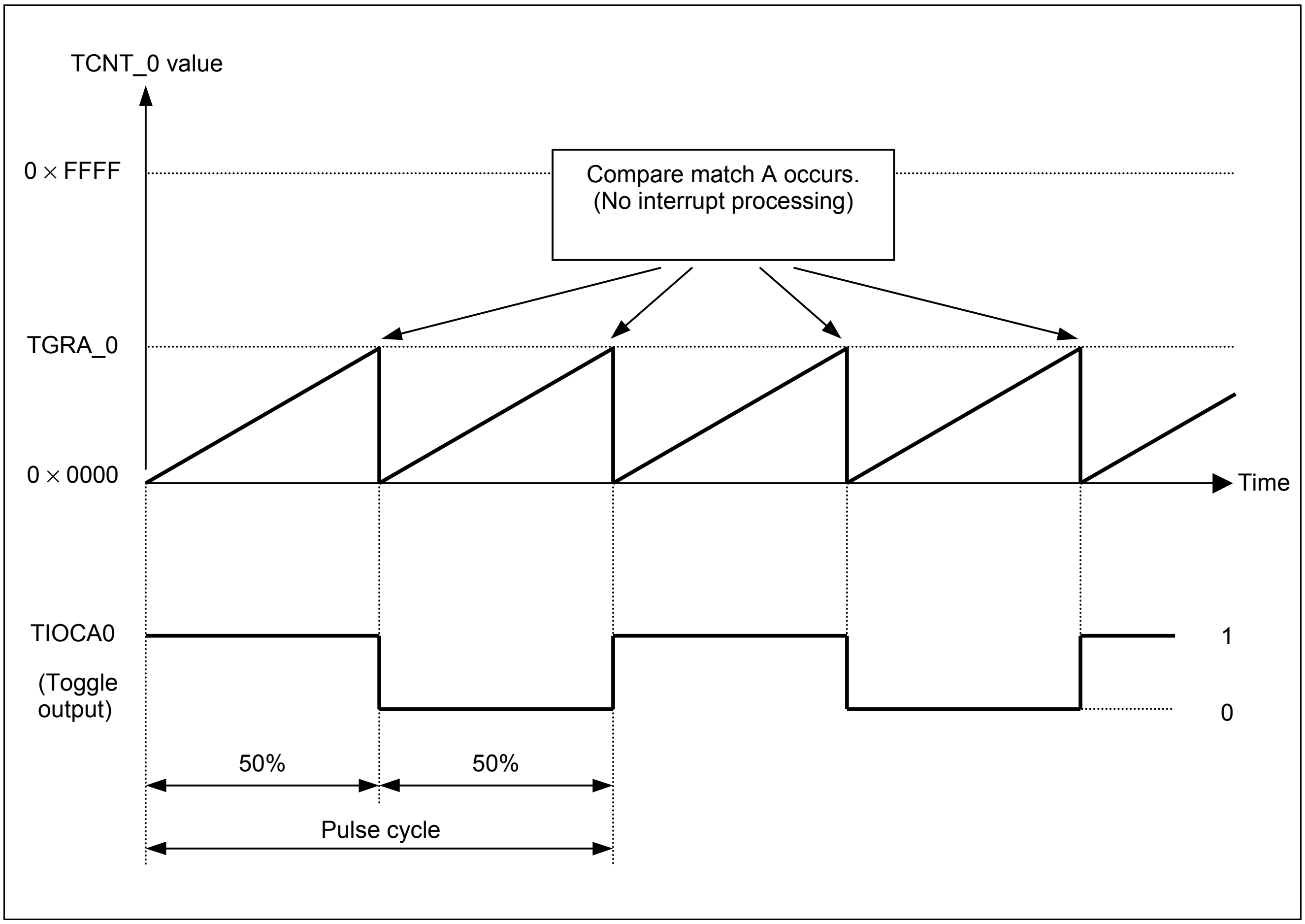


Figure 2 Example of Pulse Output

Table 1 lists the function allocations of channel 0 of the 16-bit timer pulse unit (TPU) being used.

Table 1 Function Allocation of TPU Channel 0

Type	Name	Function
Register	MSTPCRA	Cancels the TPU module stop mode.
	TSTR	Specifies whether to start or stop the timer count of TPU channel 0.
	TCR_0	Sets the TCNT_0 count clock and counter clear factor.
	TIORH_0	Sets the output pulse level when compare match A occurs.
	TGRA_0	Sets the counter value of compare match A.
Output pin	TIOCA0	Compare match A output pin

3.2 Function Specifications

The function that sets pulse output is provided as a sample program. The function's specifications are listed below.

```
void  pulse_set ( unsigned short  timer_count )
```

Argument	Description
timer_count	Specifies the timer value for the pulse cycle. The valid data range is between 0 × 0001 and 0 × FFFF. If 0 × 0000 is specified, normal operation is not performed. The count clock is fixed to Pϕ.

Return value	Description
None	—

Example)

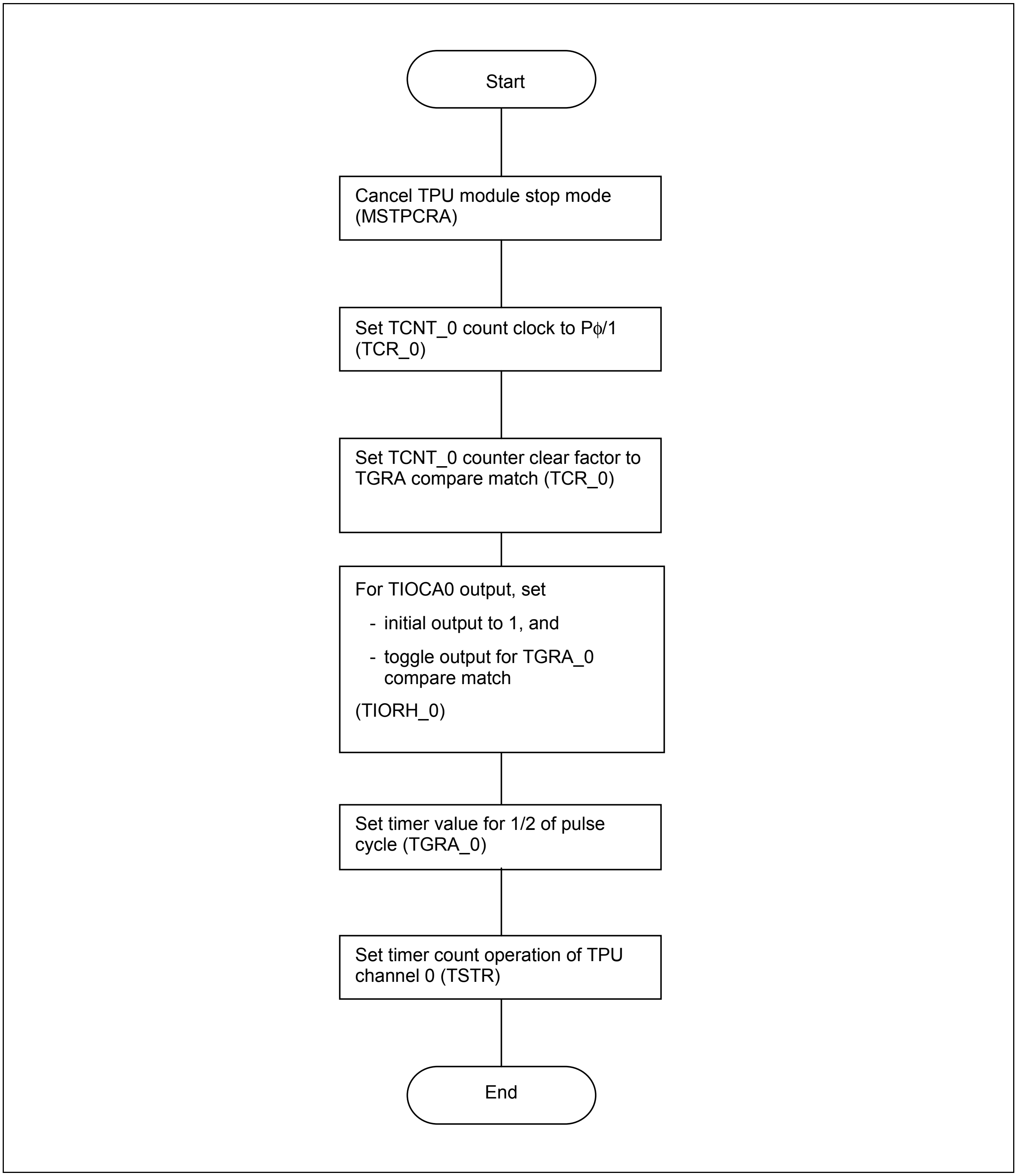
```
#define PULSE_CYCLE  1000                    // Pulse cycle:  1000 µsec (1 msec)
#define P_CLOCK                25            // Pϕ (MHz)
extern void  pulse_set ( unsigned short  ); //  External function reference declaration
void   main( void )                          //  Main routine
{
unsigned short  timer_count;                 // Calculates the timer value for the pulse cycle.
    timer_count =PULSE_CYCLE* P_CLOCK;
    pulse_set ( timer_count );               // Sets the pulse output.

 . .

}
```

3.3 Flowchart

The processing flow is shown below.



3.4 Program Listing

A listing of the source program is given below. In this source program, Renesas's standard definition (file automatically generated by High-performance Embedded Workshop: iodefine.h) is used to define the I/O register structure. To specify your own definition, change the I/O register structure in the sample program.

```
/****************************************************************/
/* include file                                                 */
/****************************************************************/
#include <machine.h>
#include "iodefine.h"

/****************************************************************/
/* function prototype                                           */
/****************************************************************/
void pulse_set( unsigned short );

/****************************************************************/
/* function definition                                          */
/****************************************************************/
void pulse_set( unsigned short timer_count )
{
    P_MSTPCRA.BIT.MSTPA0 = 0;    // reset module-standby for TPU
    P_TPU0.TCR.BIT.TPSC  = 0;    // set TPU0 countup clock source
    P_TPU0.TCR.BIT.CCLR  = 1;    // set TPU0 counter clear cause
                                 // set TPU0 compare-match-A
    P_TPU0.TIOR.BIT.IOA  = 7;                   // toggle-output
    P_TPU0.TGRA = (unsigned int)timer_count/2;  // compare value
    P_TPU.TSTR.BIT.CST0  = 1;    // start TPU0
}
```

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Sept.19.03	—	First edition issued